The Vehicle Routing Problem with Time Windows and Temporal Dependencies

Hansen, Anders Dohn; Rasmussen, Matias Sevel; Larsen, Jesper

Publication date:
2009

Document Version
Publisher's PDF, also known as Version of record

The Vehicle Routing Problem with Time Windows and Temporal Dependencies

Report 1.2009

DTU Management Engineering

Jesper Larsen
Anders Dohn
Matias Sevel Rasmussen
April 2009

DTU Management Engineering
Department of Management Engineering

The Vehicle Routing Problem with Time Windows and Temporal Dependencies

Anders Dohn, Matias Sevel Rasmussen, Jesper Larsen
{adohn,mase,jesla}@man.dtu.dk
Department of Management Engineering,
Technical University of Denmark,
Kgs. Lyngby, Denmark

May 25, 2009

Abstract

In this paper, we formulate the vehicle routing problem with time windows and temporal dependencies. The problem is an extension of the well studied vehicle routing problem with time windows. In addition to the usual constraints, a scheduled time of one visit may restrain the scheduling options of other visits. Special cases of temporal dependencies are synchronization and precedence constraints. Two compact formulations of the problem are introduced and the Dantzig-Wolfe decompositions of these formulations are presented to allow for a column-generation-based solution approach. Temporal dependencies are modeled by generalized precedence constraints. A total of four different master problem formulations are proposed and it is shown that the formulations can be ranked according to the tightness with which they describe the solution space. A tailored time window branching is used to enforce feasibility on the relaxed master problems. Finally, a computational study is carried out to quantitatively reveal strengths and weaknesses of the proposed formulations. It is concluded that, depending on the problem at hand, the best performance is achieved either by relaxing the generalized precedence constraints in the master problem, or by using a time-indexed model, where generalized precedence constraints are added as cuts when they become severely violated.

Keywords: vehicle routing with time windows; temporal dependency; generalized precedence constraints; time window branching; relaxation; column generation; branch-and-price; branch-and-cut-and-price; set partitioning; set covering; integer programming.

1 Introduction

The vehicle routing problem with time windows and temporal dependencies (VRPTWTD) is an extension of the vehicle routing problem with time windows (VRPTW). Given is a fixed set of customers with individual demands and with time windows specifying when each customer accepts service. The objective is to find routes for a number of vehicles,

all starting and ending at a central depot in such a way that the total distance is minimized. The extension that we present here is concerned with temporal dependencies between customers. A temporal dependency which is often encountered in practical instances and that has received the most attention in the literature, is the rather strict requirement of synchronization between two visits. Synchronization on visits is also used to model rendezvous between vehicles. Other, less restrictive, dependencies are constraints on minimum overlap between visits and limits on minimum or maximum gaps between visits.

There is a vast amount of literature on VRPTW and its variants. VRPTW is known to be NP-hard (Savelsbergh, 1985), nevertheless exact solution of the problem has received a lot of attention. The most successful approach is based on a Dantzig-Wolfe decomposition (Dantzig and Wolfe, 1960) of the mathematical model using column generation in a branch-and-cut-and-price framework. The method was first proposed by (Desrochers et al., 1992). The most promising recent work is based on solution of the subproblem as an elementary shortest path problem with time windows. (Feillet et al., 2004) were the first to apply this idea and were followed by (Chabrier, 2006), (Danna and Pape, 2005), (Jepsen et al., 2008), and (Desaulniers et al., 2008) among others. The approach that we present here for VRPTWTD builds on the same idea. See (Kallehauge et al., 2005) for a recent review of the literature and a thorough description of the technique.

The motivation behind this work is the many practical applications of VRPTWTD. With the inclusion of temporal dependencies in the model, we are able to describe numerous concrete problems. As (Kilby et al., 2000) point out, there is a need for more sophisticated models for the vehicle routing problem. They mention synchronization and precedence constraints as some of the relevant extensions.

(Ioachim et al., 1999) describe a fleet assignment and routing problem with synchronization constraints. The problem is solved by column generation. A similar problem with synchronization is described by (Bélanger et al., 2006). (Rousseau et al., 2003) present the synchronized vehicle dispatching problem (SVDP), which is a dynamic vehicle routing problem with synchronization between vehicles. Constraint programming and local search are applied to arrive at high-quality feasible solutions. (Lim et al., 2004) and (Li et al., 2005) study a problem from the Port of Singapore, where technicians are allocated to service jobs. For each job, a certain combination of technicians with individual skills is needed. The technicians must be present at the same time, and hence the schedule for each technician must respect a number of synchronization constraints with other schedules. The problem is solved using metaheuristics. Another application with synchronization between visits is in ground handling at airports. Teams drive around at the airport and are assigned tasks on the parked aircrafts. (Dohn et al., 2009a) describe this setup and present exact solutions to the instances considered. (Oron et al., 2008) consider ground handling with synchronization constraints as well, and present computational results for a tailored heuristic applied to data instances from an in-flight caterer in Malaysia. (Bredström and Rönnqvist, 2007) present another application of vehicle routing with synchronization constraints. A branch-and-price algorithm is applied to a realistic home care routing problem and yields promising results.

The generalization of synchronization to other temporal dependencies has been described for a few applications. (Lesaint et al., 1998) present a workforce scheduling software from a practical perspective. In the problem described, both synchronization and various other sequencing constraints occur. (Fügenschuh, 2006) describes a problem in school bus routing. Busses must wait for each other at various intermediate stops and hence precedence relations are introduced for such stops. Fügenschuh refers to the problem as the vehicle routing problem with coupled time windows. (Doerner et al., 2008) describe an application in blood collection from satellite locations for a central blood bank. Multiple visits at each location have to be scheduled with a certain slack between them. They refer to the vehicle problem as having interdependent time windows. (Bredström and Rönnqvist, 2008) modeled temporal dependencies for a home care routing problem in a mixed integer programming model (MIP) which was solved with a standard MIP solver. In (Justesen and Rasmussen, 2008) and (Dohn et al., 2008) a similar application is described and solved using branch-and-price and (Bredström and Rönnqvist, 2008) have also continued their work in this direction. An application with general temporal dependencies in machine scheduling is described by (Van Den Akker et al., 2006). Column generation is used to solve the problem. The pricing problem is primarily solved heuristically by local search and occasionally to optimality using a standard solver on an integer programming formulation of the pricing problem.

The contribution of this paper is the generalization of synchronization to any temporal dependency that can be described by generalized precedence constraints, as well as the inclusion of this in a branch-and-price context. We prove that the generalization is as strong as the formerly introduced model with synchronization. The use of the time-indexed model in the column generation is novel as well. Finally, we introduce a new set of context-free benchmark instances which enables a thorough quantitative analysis and which we hope will facilitate future research in this area.

The paper is organized as follows. In Section 2, we present two valid compact formulations of VRPTWTD. Possible decompositions of the compact formulations are presented and compared in Section 3, and for these, a tailored branching method is required. This is described in Section 4. A set of test instances are introduced in and the test results are found in Section 5. Finally, we conclude on our findings and discuss possible areas for future research in Section 6.

2 Model

In the following, we present two valid models for VRPTWTD, namely a mixed-integer model and a time-indexed model. The mixed-integer model is an extension of the model commonly used for VRPTW, whereas a time-indexed model has not received the same amount of attention.

2.1 Mixed-integer model

In the traditional vehicle routing problem with time windows, the objective is to find the shortest total travel distance to a set, $\mathcal{C}$, of n customers. Given is a fleet of identical

vehicles, $\mathcal{V}$, which are located at a central depot. Typically, the depot is represented as two locations, namely a start depot, 0, and an end depot, $n+1$. Together with the locations of all customers, they form the set, $\mathcal{N}$. All vehicles have a capacity of q. Each customer, i, has a demand, d_i, and a time window, where it accepts service $[\alpha_i, \beta_i]$. α_i is the first possible service time. The vehicle is allowed to arrive before this time, but must then wait at the customer for the time window to open. β_i is the latest possible time of initiation at customer i. $[\alpha_0, \beta_0]$ denotes the scheduling horizon of the problem. Vehicles start at the depot at time α_0 and must return to the end depot no later than β_0. τ_{ij} gives the travel time between any two locations, i and j. This may include service time at customer i. Traveling between the two locations also incurs a certain cost given by c_{ij}. We assume that q, d_i, α_i, β_i, and c_{ij} are non-negative integers and that τ_{ij} is a positive integer, respecting the triangular inequality.

The mathematical model for VRPTW is presented below. x_{ijk} is a binary variable with $x_{ijk} = 1$, if vehicle k drives directly from location i to location j, $x_{ijk} = 0$, otherwise. s_{ik} is a continuous variable and is defined as the service time at customer i, if the customer is serviced by vehicle k. Otherwise, s_{ik} is set to 0. Without restricting the model, we can fix $s_{0k} = \alpha_0, \forall k \in \mathcal{V}$ and $s_{n+1,k} = \beta_0, \forall k \in \mathcal{V}$.

$$\min \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} \sum_{k\in\mathcal{V}} c_{ij} x_{ijk} \tag{1}$$

$$\sum_{j\in\mathcal{N}} \sum_{k\in\mathcal{V}} x_{ijk} = 1 \qquad \forall i \in \mathcal{C} \tag{2}$$

$$\sum_{i\in\mathcal{C}} d_i \sum_{j\in\mathcal{N}} x_{ijk} \leq q \qquad \forall k \in \mathcal{V} \tag{3}$$

$$\sum_{j\in\mathcal{N}} x_{0jk} = 1 \qquad \forall k \in \mathcal{V} \tag{4}$$

$$\sum_{i\in\mathcal{N}} x_{ihk} - \sum_{j\in\mathcal{N}} x_{hjk} = 0 \qquad \forall h \in \mathcal{C}, \forall k \in \mathcal{V} \tag{5}$$

$$\sum_{i\in\mathcal{N}} x_{i,n+1,k} = 1 \qquad \forall k \in \mathcal{V} \tag{6}$$

$$s_{ik} + \tau_{ij} - M(1 - x_{ijk}) \leq s_{jk} \qquad \forall i, j \in \mathcal{N}, \forall k \in \mathcal{V} \tag{7}$$

$$\alpha_i \sum_{j\in\mathcal{N}} x_{ijk} \leq s_{ik} \leq \beta_i \sum_{j\in\mathcal{N}} x_{ijk} \qquad \forall i \in \mathcal{C}, \forall k \in \mathcal{V} \tag{8}$$

$$x_{ijk} \in \{0, 1\} \qquad \forall i, j \in \mathcal{N}, \forall k \in \mathcal{V} \tag{9}$$

$$s_{ik} \in \mathbb{R} \qquad \forall i \in \mathcal{N}, \forall k \in \mathcal{V} \tag{10}$$

The objective is to minimize the total cost of all edges traveled (1). All customers must be visited by exactly one vehicle (2) and the route for each vehicle must respect the capacity of that vehicle (3). (4) and (6) ensure that each route starts and ends at the depot. We also need to ensure that routes are not segmented, i.e. if a vehicle arrives at a location, it eventually leaves that location again (5). If a vehicle is set to travel between two customers, there has to be enough time between the two visits (7). Finally,

we need to make sure that all time windows are respected (8). (8) also ensures that $s_{ik} = 0$ when vehicle k does not visit customer i. (9) is the integrality constraint on x_{ijk} and (10) sets the domain of s_{ik}.

In VRPTWTD, we furthermore have a number of temporal dependencies between visits. We are able to express all of these by generalized precedence constraints. We introduce the parameter δ_{ij} which specifies the minimum difference in time from visit i to visit j. The set Δ defines all customer pairs (i, j) for which a temporal dependency exists. The generalized precedence constraints are formulated as follows, where $\sum_{k \in \mathcal{V}} s_{ik}$ is the start time of visit i.

$$\sum_{k \in \mathcal{V}} s_{ik} + \delta_{ij} \leq \sum_{k \in \mathcal{V}} s_{jk} \qquad \forall (i, j) \in \Delta \tag{11}$$

Constraint (11) can be used to model all the temporal dependencies that were observed in the literature review. There may be dependencies between several visits, e.g. synchronization of three or more visits. Such dependencies are modeled by applying the corresponding pairwise dependencies. In this paper, we will focus on five kinds of temporal dependencies that are commonly found in practice. These are visualized in Figure 1.

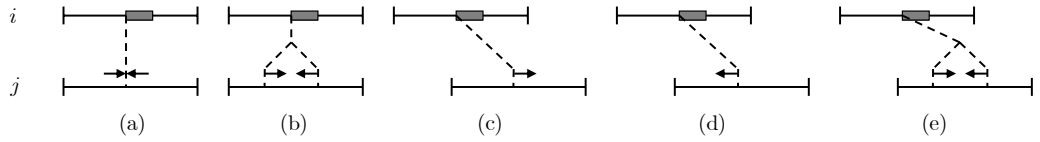

Figure 1: Five kinds of temporal dependencies that are often encountered in practice. Each of the five subfigures shows the time windows of two visits i and j with a temporal dependency between them. Assuming some start time for visit i, the dashed line together with the arrows give the corresponding feasible part of the time window of visit j. (a) synchronization, (b) overlap, (c) minimum difference, (d) maximum difference, (e) minimum+maximum difference.

It is straight forward to model the temporal dependencies of Figure 1 using constraint (11). The correct values for δ_{ij} and δ_{ji} are listed in Table 1.

2.2 Time-indexed model

Time-indexed formulations have not received much attention in the column generation context of VRPTW. A time-index formulation is usually disregarded because of its vast size. It is, however, popular in the formulation of machine scheduling problems, as it gives a tight description of precedence constraints. Here, we present the time-indexed model of VRPTWTD, as it will be used to strengthen the bounds in the branch-and-price algorithm. We introduce the index $t \in \mathcal{T}$ on the x-variable, with $\mathcal{T} = \{\alpha_0, \ldots, \beta_0\}$. x_{ijkt} is defined as: $x_{ijkt} = 1$, if vehicle k services customer i at time t and then drives

	Temporal dependency	δ_{ij}	δ_{ji}
(a)	Synchronization	0	0
(b)	Overlap	$-\text{dur}_j$	$-\text{dur}_i$
(c)	Minimum difference	$\text{diff}_{\min}$	N/A
(d)	Maximum difference	N/A	$-\text{diff}_{\max}$
(e)	Minimum+maximum difference	$\text{diff}_{\min}$	$-\text{diff}_{\max}$

Table 1: Parameter values for the five temporal dependencies of Figure 1. dur_i is the service time at customer i.

directly to location j. $x_{ijkt} = 0$, otherwise.

$$\min \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} \sum_{k\in\mathcal{V}} \sum_{t\in\mathcal{T}} c_{ij} x_{ijkt} \tag{12}$$

$$\sum_{j\in\mathcal{N}} \sum_{k\in\mathcal{V}} \sum_{t\in\mathcal{T}} x_{ijkt} = 1 \qquad \forall i \in \mathcal{C} \tag{13}$$

$$\sum_{i\in\mathcal{C}} d_i \sum_{j\in\mathcal{N}} \sum_{t\in\mathcal{T}} x_{ijkt} \leq q \qquad \forall k \in \mathcal{V} \tag{14}$$

$$\sum_{j\in\mathcal{N}} \sum_{t\in\mathcal{T}} x_{0jkt} = 1 \qquad \forall k \in \mathcal{V} \tag{15}$$

$$\sum_{i\in\mathcal{N}} \sum_{t\in\mathcal{T}} x_{ihkt} - \sum_{j\in\mathcal{N}} \sum_{t\in\mathcal{T}} x_{hjkt} = 0 \qquad \forall h \in \mathcal{C}, \forall k \in \mathcal{V} \tag{16}$$

$$\sum_{i\in\mathcal{N}} \sum_{t\in\mathcal{T}} x_{i,n+1,kt} = 1 \qquad \forall k \in \mathcal{V} \tag{17}$$

$$\sum_{t'=t,\ldots,\beta_0} x_{ijkt'} + \sum_{t'=\alpha_0,\ldots,\min(\beta_0,t+\tau_{ij}-1)} x_{jhkt'} \leq 1 \qquad \forall i,j,h \in \mathcal{N}, \forall k \in \mathcal{V}, \forall t \in \mathcal{T} \tag{18}$$

$$\sum_{t'=t,\ldots,\beta_0} x_{ihkt'} + \sum_{t'=\alpha_0,\ldots,\min(\beta_0,t+\delta_{ij}-1)} x_{jglt'} \leq 1 \qquad \forall (i,j) \in \Delta, \forall h,g \in \mathcal{C}, \forall k,l \in \mathcal{V}, \forall t \in \mathcal{T} \tag{19}$$

$$x_{ijkt} = 0 \qquad \begin{array}{l} \forall i \in \mathcal{C}, j \in \mathcal{N}, \forall k \in \mathcal{V}, \\ \forall t \in \{\alpha_0, \ldots, \alpha_i - 1\} \cup \{\beta_i + 1, \ldots, \beta_0\} \end{array} \tag{20}$$

$$x_{ijkt} \in \{0,1\} \qquad \forall i,j \in \mathcal{N}, \forall k \in \mathcal{V}, \forall t \in \mathcal{T} \tag{21}$$

Constraints (12)–(17) are similar to (1)–(6), where we now sum over the time index as well. (18) provides the required travel time between visits. The strength of this model lies in the formulation of generalized precedence constraints (19). (19) is the equivalent of (11) of the former model. The constraint states that if visit i is scheduled anywhere from time t and onward, then visit j is not scheduled before time $t + \delta_{ij}$. This is valid for all $t \in \mathcal{T}$. (20) enforces the time windows and (21) is the integrality constraint.

3 Decomposition

As described earlier, Dantzig-Wolfe decomposition has been very successful in exact optimization of VRPTW. The decomposition splits the problem into a set-partitioning master problem and a constrained shortest path subproblem. See e.g. (Kallehauge et al., 2005) for a thorough exposition. In the traditional VRPTW formulation, (2) are the only constraints that link the vehicles. Without this, we can solve the problem separately for each vehicle. Hence, the problem is split into a subproblem, where feasible routes are generated and a master problem, where these routes are combined.

3.1 Master problem

We propose four applicable formulations of the master problem and rank them according to the tightness with which they describe the solution space.

3.1.1 Mixed-integer formulation

The introduced generalized precedence constraints apply to routes from separate vehicles, and hence these will be part of the new master problem. In the full master problem, we have the set of all feasible routes, $\mathcal{R}$. Each route is defined by the customers visited and the time of each such visit, described by two parameters, a_i^r and s_i^r. For each route, r, and each customer, i, if customer i is in the route r, we set $a_i^r = 1$ and s_i^r equal to the time of that visit, and if the customer is not in the route, $a_i^r = 0$ and $s_i^r = 0$. In column generation, the master problem variables are generated iteratively and the set of variables available in a specific iteration is denoted $\mathcal{R}'$. Decision variables for the master problem are denoted λ_r, with $\lambda_r = 1$, if route r is used, and $\lambda_r = 0$, otherwise. The LP-relaxation of the master problem defined by a subset of the decision variables, $\mathcal{R}'$, is denoted the *restricted master problem* and is formulated below.

$$\min \sum_{r \in \mathcal{R}'} c_r \lambda_r \tag{22}$$

$$\sum_{r \in \mathcal{R}'} a_i^r \lambda_r = 1 \qquad \forall i \in \mathcal{C} \tag{23}$$

$$\sum_{r \in \mathcal{R}'} s_i^r \lambda_r + \delta_{ij} \leq \sum_{r \in \mathcal{R}'} s_j^r \lambda_r \qquad \forall (i,j) \in \Delta \tag{24}$$

$$\lambda_r \geq 0 \qquad \forall r \in \mathcal{R}' \tag{25}$$

The corresponding subproblem is that of generating negative reduced cost routes for (22)–(25). In this context, we refer to the model as the *mixed-integer formulation*. The main disadvantage of the model is that it introduces linear time costs in the subproblem, namely the dual variables of constraints (24). Hence, the subproblem is a shortest path problem with time windows and linear node costs. Another issue is that s_i^r is a non-binary parameter, and the introduction of non-binary parameters in the master problem is usually a feature that leads to highly fractional solutions.

3.1.2 Time-index formulation

In the time-index formulation, the master problem contains only binary parameters. (13) and (19) link the vehicles and must therefore remain in the master problem. The parameters of the time-indexed master problem are defined as $a_{it}^r = 1$ if customer i is scheduled at time t in route r, and $a_{it}^r = 0$ otherwise. The decision variable λ_r has the same definition as in the previous model. The restricted master problem of the *time-indexed formulation* is:

$$\min \sum_{r \in \mathcal{R}'} c_r \lambda_r \tag{26}$$

$$\sum_{r \in \mathcal{R}'} \sum_{t' \in \mathcal{T}} a_{it'}^r \lambda_r = 1 \qquad \forall i \in \mathcal{C} \tag{27}$$

$$\sum_{r \in \mathcal{R}'} \sum_{t'=t,\ldots,\beta_0} a_{it'}^r \lambda_r + \sum_{r \in \mathcal{R}'} \sum_{t'=\alpha_0,\ldots,\min(\beta_0, t+\delta_{ij}-1)} a_{jt'}^r \lambda_r \leq 1 \qquad \forall (i,j) \in \Delta, \forall t \in \mathcal{T} \tag{28}$$

$$\lambda_r \geq 0 \qquad \forall r \in \mathcal{R}' \tag{29}$$

The obvious problem with the time-indexed master problem (26)–(29) is the number of constraints. The scheduling horizon is usually large enough to make this model intractable in realistic problems. The subproblem is a shortest path problem with time windows and time-dependent costs. The costs may be different for each time step. This is very unlikely, however. Most of the constraints of type (28) will be non-binding and this leaves the corresponding dual variables equal to 0. As most of these constraints will be non-binding, we may choose to introduce them, only when they become violated. To identify constraints which are violated, a separation algorithm is used. The separation algorithm is described in more detail in Section 3.2.

3.1.3 Relaxed formulation

A third way to approach the problem is to simply disregard the temporal dependencies in the master problem. The dependencies must then be enforced by the branching scheme. This approach is used for synchronization by (Dohn et al., 2009a) and (Bredström and Rönnqvist, 2007) and for generalized precedence constraints by (Justesen and Rasmussen, 2008). It leaves the following master problem, which is identical to the master problem of the VRPTW decomposition. Here, we refer to it as the *relaxed formulation*.

$$\min \sum_{r \in \mathcal{R}'} c_r \lambda_r \tag{30}$$

$$\sum_{r \in \mathcal{R}'} a_i^r \lambda_r = 1 \qquad \forall i \in \mathcal{C} \tag{31}$$

$$\lambda_r \geq 0 \qquad \forall r \in \mathcal{R}' \tag{32}$$

3.1.4 Time-regulating formulation

In (Dohn et al., 2009a) it is considered to introduce a binary variable that specifies the time of each visit and constraints are added to enforce synchronization. This idea transfers to the general case considered in this paper. It is, however, proved that the model considered is not stronger than the corresponding relaxed formulation. Analogously, it is easily shown that time-regulating formulation here is a relaxation of the relaxed formulation and hence it is not of great interest.

3.1.5 Strength of the formulations

The relaxed formulation is obviously a relaxation of both the mixed-integer formulation and the time-indexed formulation. An interesting result is that the mixed-integer formulation is also a relaxation of the time-indexed formulation, and we are hence able to order the four models according to their strength.

Proposition 1 *The time-indexed formulation is a stronger formulation than the mixed-integer formulation.*

Proof. Proof In the following, we assume that we have a solution to (26)–(29) and prove that the solution is also feasible for (22)–(25). For all problems with a feasible solution, it holds that $\alpha_0 + \delta_{ij} - 1 \leq \beta_0, \forall (i,j) \in \Delta$ and hence a special case of (28) with $t = \alpha_0$ is:

$$\sum_{r \in \mathcal{R}'} \sum_{t' = \alpha_0, \ldots, \beta_0} a^r_{it'} \lambda_r + \sum_{r \in \mathcal{R}'} \sum_{t' = \alpha_0, \ldots, \alpha_0 + \delta_{ij} - 1} a^r_{jt'} \lambda_r \leq 1 \qquad \forall (i,j) \in \Delta \tag{33}$$

Using (27) this entails the rather obvious:

$$\sum_{r \in \mathcal{R}'} \sum_{t' = \alpha_0, \ldots, \alpha_0 + \delta_{ij} - 1} a^r_{jt'} \lambda_r = 0 \qquad \forall (i,j) \in \Delta \tag{34}$$

$$\Rightarrow \sum_{r \in \mathcal{R}'} \sum_{t' = \alpha_0, \ldots, t} a^r_{jt'} \lambda_r = 0 \qquad \forall (i,j) \in \Delta, t = \alpha_0, \ldots, \alpha_0 + \delta_{ij} - 2 \tag{35}$$

$$\Rightarrow \sum_{r \in \mathcal{R}'} \sum_{t' \in \mathcal{T}} a^r_{it'} \lambda_r + \sum_{r \in \mathcal{R}'} \sum_{t' = \alpha_0, \ldots, t} a^r_{jt'} \lambda_r = 1 \qquad \forall (i,j) \in \Delta, t = \alpha_0, \ldots, \alpha_0 + \delta_{ij} - 2 \tag{36}$$

Summing the constraints (27), (36), and (28) over t, we get the following for $(i,j) \in \Delta$:

$$\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} a^r_{it'}\lambda_r = 1$$

$$\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} a^r_{it'}\lambda_r + \sum_{r\in\mathcal{R}'}\sum_{t'=\alpha_0,\ldots,t} a^r_{jt'}\lambda_r = 1 \quad t=\alpha_0,\ldots,\alpha_0+\delta_{ij}-2$$

$$\sum_{r\in\mathcal{R}'}\sum_{t'=t,\ldots,\beta_0} a^r_{it'}\lambda_r + \sum_{r\in\mathcal{R}'}\sum_{t'=\alpha_0,\ldots,\min(\beta_0,t+\delta_{ij}-1)} a^r_{jt'}\lambda_r \leq 1 \quad t=\alpha_0,\ldots,\beta_0$$

$$\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} (t'-\alpha_0+1+\delta_{ij})a^r_{it'}\lambda_r + \sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} (\beta_0+\delta_{ij}-t')a^r_{jt'}\lambda_r \leq \beta_0-\alpha_0+\delta_{ij}+1$$

Therefore, for any feasible solution of (26)–(29), we have for $(i,j)\in\Delta$:

$$0 \leq \beta_0-\alpha_0+\delta_{ij}+1-\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}}(t'-\alpha_0+1+\delta_{ij})a^r_{it'}\lambda_r-\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}}(\beta_0+\delta_{ij}-t')a^r_{jt'}\lambda_r \tag{37}$$

$$= \beta_0-\alpha_0+\delta_{ij}+1-\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} t'a^r_{it}\lambda_r-(-\alpha_0+1+\delta_{ij})\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} a^r_{it'}\lambda_r \tag{38}$$
$$-(\beta_0+\delta_{ij})\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} a^r_{jt'}\lambda_r+\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} t'a^r_{jt'}\lambda_r$$

$$= \beta_0-\alpha_0+\delta_{ij}+1-\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} t'a^r_{it'}\lambda_r+\alpha_0-1-\delta_{ij}-\beta_0-\delta_{ij}+\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} t'a^r_{jt'}\lambda_r \tag{39}$$

$$= \sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} t'a^r_{jt'}\lambda_r-\sum_{r\in\mathcal{R}'}\sum_{t'\in\mathcal{T}} t'a^r_{it'}\lambda_r-\delta_{ij} \tag{40}$$

$$= \sum_{r\in\mathcal{R}'} s^r_j\lambda_r-\sum_{r\in\mathcal{R}'} s^r_i\lambda_r-\delta_{ij} \tag{41}$$

The result in (39) is based on (27). The final result in (41) comes from the following relation between the parameters of the models (22)–(25) and (26)–(29): $s^r_i=\sum_{t\in\mathcal{T}} ta^r_{it}$. The result in (41) proves that any feasible solution of (26)–(29) also respects (24). (23) is trivially respected as $a^r_i=\sum_{t'\in T} a^r_{it'}$, and hence (22)–(25) is a relaxation of (26)–(29).

To illustrate that the two formulations are not equally strong, we consider the following small example. Take two customers i and j with $\delta_{ij}=2$. Three simple routes cover these two customers with $a^1_1=1, a^2_2=1, a^3_2=1$ and $s^1_1=1, s^2_2=2, s^3_2=4$ for model (22)–(25). In model (26)–(29) this corresponds to $a^1_{11}=1, a^2_{22}=1, a^3_{24}=1$. A solution with $\lambda_1=1,\lambda_2=0.5,\lambda_3=0.5$ is feasible in (22)–(25) but not in (26)–(29). This is verified by inspecting (24) for $i=1, j=2$:

$$\sum_{r\in\mathcal{R}'} s^r_1\lambda_r+\delta_{12}\leq\sum_{r\in\mathcal{R}'} s^r_2\lambda_r \Rightarrow 1+2\leq 3$$

and (28) for $i = 1, j = 2, t = 1$:

$$\sum_{r \in \mathcal{R}'} (a_{11}^r \lambda_r + a_{12}^r \lambda_r + a_{13}^r \lambda_r + a_{14}^r \lambda_r) + \sum_{r \in \mathcal{R}'} (a_{21}^r \lambda_r + a_{22}^r \lambda_r) \leq 1 \Rightarrow 1 + 0.5 \leq 1$$

■

Corollary 1 *The time-indexed formulation is a stronger formulation than the mixed-integer formulation. The mixed-integer formulation is stronger than the relaxed formulation, which in turn is at least as strong as the time-regulating formulation.*

A nice property of the time-indexed model is that it has only been relaxed with respect to integrality and this means that if we can restore integrality, we have a feasible solution. In this paper, we will only consider branching to restore integrality, and hence the advantage may not seem immediate. For VRPTW, a significant amount of work has been done on cut generation to remove fractional solutions. Such cuts could be added to the time-indexed model of VRPTWTD as well, and this may restore integrality without the need of branching. See e.g. the work of (Kohl et al., 1999), (Cook and Rich, 2001), (Lysgaard et al., 2004), and (Jepsen et al., 2008) for more on the subject.

3.2 Separation algorithm

As described earlier, the generalized precedence constraints (28) of the time-indexed master problem (26)–(29) are only represented implicitly. The constraints are added as cuts, as they become violated. The constraint is repeated below.

$$\sum_{r \in \mathcal{R}'} \sum_{t'=t,\ldots,\beta_0} a_{it'}^r \lambda_r + \sum_{r \in \mathcal{R}'} \sum_{t'=\alpha_0,\ldots,\min(\beta_0, t+\delta_{ij}-1)} a_{jt'}^r \lambda_r \leq 1 \qquad \forall (i,j) \in \Delta, \forall t \in \mathcal{T} \quad (28)$$

A separation algorithm is applied to identify violated cuts. In theory, we have to check for violations for all $t \in \mathcal{T}$, but actually it is possible to make do with significantly less. As a_{it}^r is a binary parameter and $\lambda_r \geq 0$, the sum $\sum_{r \in \mathcal{R}'} \sum_{t'=t,\ldots,\beta_0} a_{it'}^r \lambda_r$ is non-increasing for increasing t. Correspondingly, the sum $\sum_{r \in \mathcal{R}'} \sum_{t'=\alpha_0,\ldots,\min(\beta_0, t+\delta_{ij}-1)} a_{jt'}^r \lambda_r$ is non-decreasing. (28) is never violated for $t = \alpha_0$ as such violations are prevented by preprocessing the time windows, see Section 4.1. Therefore, for visit i, we only need to check for violations with any t where $\exists r : a_{it}^r \lambda_r > 0$, i.e. any point in time where visit i is scheduled (possibly with a fractional value). It is easy to generate a list of all t where $\exists r : a_{it}^r \lambda_r > 0$, by running through the routes of all variables with positive values and registering the time of each visit. By separating cuts as described, we are not adding all violated cuts, but we are sure to add at least one cut for each visit, if any cuts are violated for that visit.

3.3 Subproblem

The subproblem of the Dantzig-Wolfe decomposition of VRPTW is an elementary shortest path problem with time windows and capacity constraints (ESPPTWCC). Any feasible solution of the subproblem with negative cost represents a column with negative reduced cost in the master problem and can therefore enter the basis. The subproblem consists of constraints (3)–(10). The variables are defined as in the compact formulation, but now for the single vehicle under consideration, i.e. the index k has been removed. The objective function of the subproblem becomes:

$$\min \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} (c_{ij} - \pi_i) x_{ij} \tag{42}$$

(Dror, 1994) proves that ESPPTW is NP-hard in the strong sense and hence no pseudo-polynomial algorithms are likely to exist. The subproblem is usually solved with a dynamic label setting algorithm. (Desrochers et al., 1992) presented a dynamic algorithm for the non-elementary version of the subproblem. This algorithm was adjusted to handle the elementary problem by (Feillet et al., 2004) and superior results based on this method have been presented recently, see e.g. (Desaulniers et al., 2008). The idea in the label setting algorithm is that a set of partial paths are represented by labels. Given a label for some partial path, it is possible to expand the path by creating new labels in nodes that can possibly extend the current partial path. The length of the path is increased by one, and the process continues iteratively.

The subproblem of the mixed-integer formulation must also consider the dual variables of constraint (24), σ_{ij}, and hence the objective function becomes:

$$\min \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} (c_{ij} - \pi_i)\, x_{ij} - \sum_{(i,j) \in \Delta} \sigma_{ij} s_i + \sum_{(j,i) \in \Delta} \sigma_{ji} s_i \tag{43}$$

As described previously, the subproblem is now a shortest path problem with time windows and linear node costs, which makes it much harder to solve. (Ioachim et al., 1998) describe a dynamic algorithm to solve the acyclic version of this problem. A similar cyclic problem is solved as a subproblem by (Christiansen and Nygreen, 2005).

The subproblem of the time-indexed formulation has the following objective function:

$$\min \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} (c_{ij} - \pi_i) \sum_{t \in \mathcal{T}} x_{ijt} - \sum_{(i,j) \in \Delta} \sum_{t \in \mathcal{T}} \sum_{t' = \alpha_0, \ldots, t} \rho_{ijt'} x_{ijt} - \sum_{(j,i) \in \Delta} \sum_{t' = \max(\alpha_0, t - \delta_{ji} + 1), \ldots, \beta_0} \rho_{jit'} x_{ijt} \tag{44}$$

where ρ_{ijt} is the non-positive dual variables of constraint (28). In the worst case, this objective function introduces a distinct cost for each time step. In a label setting algorithm this means that we have to create one label for each time step and using this approach, the number of labels explodes immediately. In practice, only a few constraints of type (28) are binding and hence only few ρ_{ijt} have non-zero values.

The idea in the basic label setting algorithm is to only create and keep labels that are not dominated by another better label. With the objective function (44), we have a

lot of potential labels. It is only profitable to postpone a visit, if it can possibly decrease the objective. As $\rho_{ijt} \leq 0$ and as ρ_{ijt} is always subtracted, this is only possible, if we can avoid terms in the sum. Therefore, for visit i, we only need a label for the earliest possible time t^0 and one label for each t in $\{t^0, \ldots, \beta_i\}$, where $\exists (j,i) \in \Delta, \exists t' \in \mathcal{T}$ with $t' = t - \delta_{ji} \wedge \rho_{jit'} < 0$. For all other potential labels, there will always be a label earlier in time with the same or less cost.

A small improvement, that we found to have a significant effect, is to include knowledge of mutually exclusive visits. Some temporal dependencies like e.g. synchronization and overlap make it impossible to include both visits in the same route. This restriction is imposed in the master problem or in the branching scheme. Hence, routes could be generated that would never occur in a feasible solution. By disallowing mutually exclusive visits to occur in all routes generated in the subproblem, the LP-bounds get stronger and as a consequence the algorithm is more effective.

4 Branching

The master problem models presented in the previous section are relaxations and therefore we may need to carry out branching to get to a feasible solution. In the mixed-integer formulation and the time-indexed formulation, the integer property of the λ-variables has been relaxed and therefore integrality needs to be restored by branching. A lot of work has already been done for VRPTW in this respect. See e.g. Kallehauge et al. (2005) for a review. In the relaxed formulation, the generalized precedence constraints have also been relaxed. Therefore, in this model, we need a branching method that will also restore feasibility with respect to temporal dependencies.

Gélinas et al. (1995) proposed to branch on time variables in order to arrive at integer-feasible solutions. This type of branching was also used to enforce synchronization by Ioachim et al. (1999), Dohn et al. (2009a), and Bredström and Rönnqvist (2007), and for general temporal dependencies by Justesen and Rasmussen (2008). Time window branching is not complete with respect to integer feasibility and hence has to be complemented by another branching scheme, e.g. the traditional branching on a flow variable.

4.1 Time window reduction

Before describing the actual branching scheme, we introduce a simple reduction technique based on the generalized precedence constraints. For any two visits, i and j with $(i,j) \in \Delta$, it is possible to reduce the time windows as follows:

	Visit i	Visit j
Old time windows	$[\alpha_i, \beta_i]$	$[\alpha_j, \beta_j]$
New time windows	$[\alpha_i, \min(\beta_i, \beta_j - \delta_{ij})]$	$[\max(\alpha_j, \alpha_i + \delta_{ij}), \beta_j]$

These reductions are also illustrated in Figure 2. The reductions are used to preprocess the time windows and may also be used anywhere in the branching tree. This is useful when applying time window branching.

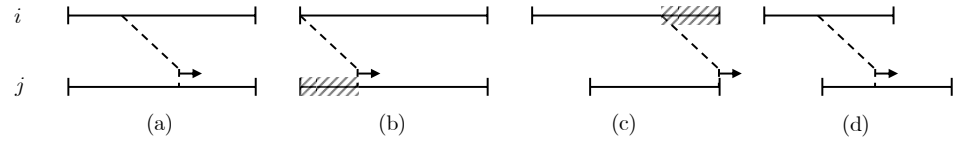

Figure 2: Time window reductions. (a) The original time windows. (b) Using the generalized precedence constraint, the first part of the time window of visit j is removed. (c) In a similar way, the last part of the time window of visit i is removed. (d) The time windows after the reduction.

4.2 Time window branching

In a feasible solution of VRPTWTD, all visits are scheduled at exactly one point in time and all generalized precedence constraints are respected. In the relaxed formulation, a solution may be integer feasible but could still violate precedence constraints. In the two other models, an integer feasible solution will also respect precedence constraints. As for VRPTW, we may still use time window branching to get integral solutions. In the following, we use the relaxed formulation as a basis for introducing time window branching, but it transfers easily to the other models.

Figure 3 shows a violation of the precedence constraint between visits i and j, in routes r_2 and r_1. By branching on the time window of visit i and using the time window reduction rule of Section 4.1 for (j, i), r_1 and r_2 are prohibited in the left and right branch, respectively. Note that there is no overlap between the time window of visit i in the left branch and the corresponding time window in the right branch.

As long as the split time is chosen somewhere between s_i and $s_j + \delta_{ji}$, the current solution will be excluded from the solution space by using the reduction rule for (j, i). The tightest formulation is reached if time windows are reduced as much as possible. Therefore, with the new time window of visit i, we run through all relevant precedence constraints and reduce time windows where possible. This may also reduce the time windows of other visits than i and j, and this process is repeated iteratively, until no further reduction is possible.

An interesting result is that this branching strategy is as strong as the one formerly proposed specifically for synchronization. In the less general context, the time windows of two synchronized visits are, naturally, always identical. Branching is done on the two time windows simultaneously, so they always stay identical. Synchronization modeled by two generalized precedence constraints, also has this property when time window reductions are applied. This is illustrated in Figure 4. The time windows of i and j are identical in each of the branches after applying time window reduction.

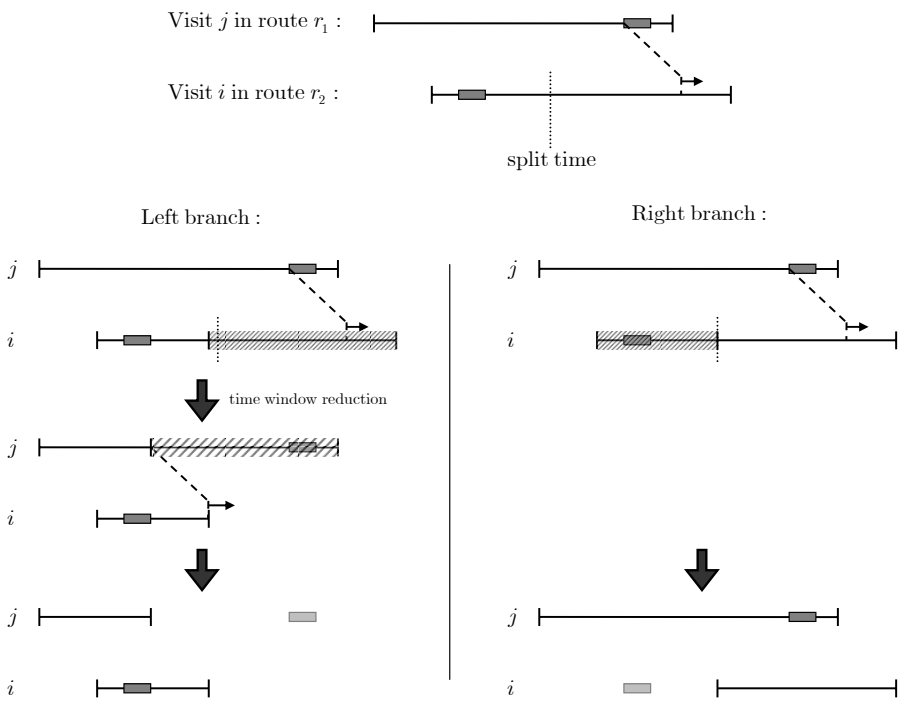

Figure 3: Branching to avoid a violation of a precedence constraint.

Usually, there are several branching candidates to choose from and we need a strategy to choose one of these. Gélinas et al. (1995) elaborate further on this subject. When using strong branching (see e.g. Achterberg et al. (2005)) a few candidates are chosen for further probing. In any case, we need to specify a priority ordering of candidates. First, we need to find the potential branching candidates. In theory, we could branch on any time window and split it at an arbitrary position. In practice, however, we limit this choice. We do not want to consider candidates where the branching is without effect in one of the branches, i.e. where one of the branches does not prohibit any columns of the current solution. Also, many of the remaining candidates have an identical effect on the current solution. They may still have a different effect on new columns, but it is very hard to predict this impact. Figure 5 (a) depicts some of the potential branching candidates in the time window of visit i. Visit i is a part of several routes that have been included in the solution with fractional values and hence it appears at multiple positions within its own time window. In this example we assume that the routes r_2 and r_3 are both in the solution with a value of 0.5 and r_1 and r_4 with a value of 1. The effect on the current solution of each candidate is shown in Table 2. Candidates 1 and 6 are examples of ineffective candidates. Candidates 2 and 3 have an identical effect on the current solution.

In our approach, when choosing between candidates with an identical immediate effect, we select the candidate which splits at the latest possible time. This choice is

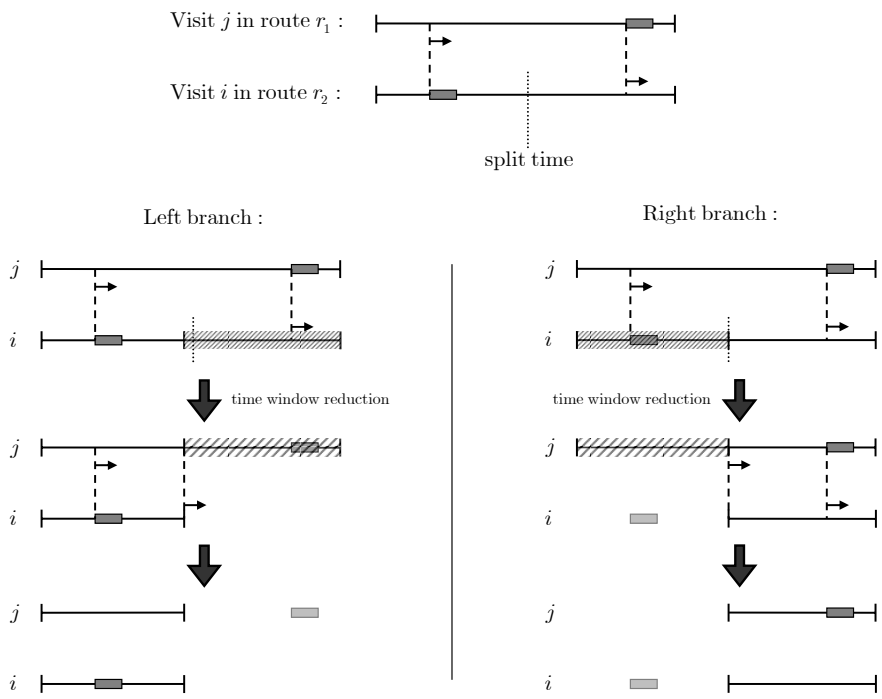

Figure 4: Branching on a generalized precedence constraint of a synchronization constraint.

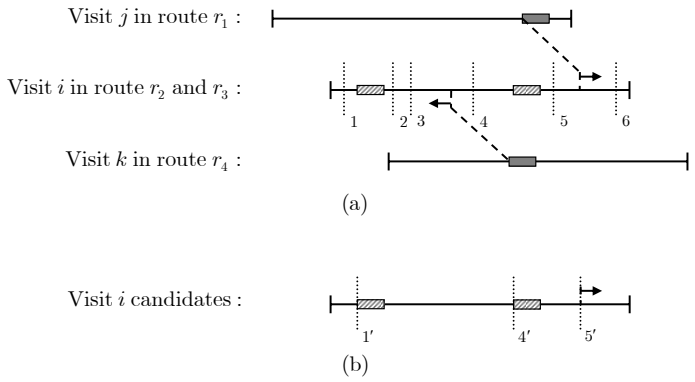

Figure 5: Some potential branching candidates in the time window of visit i.

made due to the following algorithmic considerations: 1) We do not want a route that has been prohibited in the branch to get a small fix and then reappear in the solution, and 2) The candidate should at least need to reroute some of the visits, and hence create a larger diversity between the branches. The label setting algorithm used to generate routes will always try to schedule visits as early as possible, when everything else is

	Infeasible routes		Sum of excluded variables		
	Left branch	Right branch	Left branch	Right branch	Preference
Candidate 1	r_1, r_2, r_3		2	0	0
Candidate 2	r_1, r_3	r_2	1.5	0.5	0.5
Candidate 3	r_1, r_3	r_2	1.5	0.5	0.5
Candidate 4	r_1, r_3	r_2, r_4	1.5	1.5	1.5
Candidate 5	r_1	r_2, r_3, r_4	1	2	1
Candidate 6		r_2, r_3, r_4	0	2	0

Table 2: Effect of the branching candidates of Figure 5.

equal. This means that there is a good chance that visits can be moved to a slightly later position, without disrupting the route. On the other hand, it is impossible to schedule the visit earlier without rerouting. The separation routine presented in Section 3.2 similarly benefits from this rationale.

Turning back to Figure 5 (a), we prefer candidate 3 to candidate 2 as there is less chance that r_2 can be adapted to the new time window of the right branch. We prefer candidate 4 to candidate 3 as it excludes the same or more in both branches. Figure 5 (b) shows the three candidates that we would actually consider for the time window of visit i. The candidates $1'$, $4'$, and $5'$ get the same values as 1, 4, and 5, respectively, in Table 2. Remember that these are just the candidates of visit i. There will be similar candidates for each of the other visits. We find the candidates for visit i by running through all routes that are included in the solution with a positive value. If visit i is in the route, the start time of the visit in that route is a candidate. If the route includes a visit j, where $(j, i) \in \Delta$ the route contributes with a candidate for visit i with split time $s_j + \delta_{ji}$.

In this paper, the problems are solved to optimality, which means that every node in the branch-and-bound tree must be either explored or pruned. Hence, we aim for a small, but at the same time, balanced tree. To achieve this, we rank the branching candidates according to the corresponding sums of excluded variables in the two branches. A candidate gets the value of the minimum of the two sums, and hence only the worst of the branches counts. A larger value of a candidate is equal to a high preference. This means that we prefer branching candidates which exclude as much as possible in the branch, where they exclude the least. In the example, candidate 4 is preferred, as it excludes 1.5 routes in each branch, giving it a preference ranking of 1.5.

If the aim is to get high-quality solutions, but not necessarily optimal solutions, in a short time, it may be better to choose branching candidates where one of the branches is more promising than the other. This may then be utilized in a heuristic search of the branch-and-bound tree. This idea has been used in several other contexts, see e.g. Ryan (1992).

5 Test results

The intention of this section is to give a general overview of the complexity of vehicle routing problems with temporal dependencies. The tests are summarized in graphs that capture the trends we see in the tests overall.

A set of benchmark instances have been used in the following quantitative analysis. The generation of these instances is explained in detail in the technical report by Dohn et al. (2009b). The instances are extensions of the 56 well known VRPTW-instances of Solomon (1987). Solomon's VRPTW-instances have been used extensively in existing literature and new solution algorithms for VRPTW are usually tested on these to indicate how the algorithm performs. The data sets consist of a number of customers with a geographical location, a time window, and a demand. The instances are publicly available. We take the original instances and introduce temporal dependencies of various types to these instances. We have chosen to look only at the instances with 25 customers, as these are small enough to allow quick solution of the basic problem. Some of them still prove hard to solve as temporal dependencies are introduced. From preliminary tests it was clear that instances with 50 customers proved too hard to solve. This corresponds well with the results in the literature, where some of the 50 customer problems without temporal dependencies remained unsolved until very recently.

Five sets of instances were made and in this section, we have chosen to focus on two of the instance sets, namely instances with only synchronization relations and a set with a random mix of the five temporal dependencies of Table 1. These have been chosen since the first represents a large group of practical applications and the latter does not hold any particular structure. The statements made in the following are in full accordance with the other instance sets as well.

The algorithms are implemented in the branch-and-cut-and-price framework of COIN-OR Lougee-Heimer (2003); Coin (2006). The tests have been run on 2.2 GHz AMD processors with 2 GB RAM. Based on preliminary tests, the algorithm is set to do strong branching with three candidates and adds up to five variables with negative reduced cost per iteration. For as long as possible, columns are generated by a heuristic version of the label setting algorithm similar to the one proposed by Chabrier (2006).

As will become apparent in the following, both the time-indexed formulation and the relaxed formulation have areas where one does better than the other. Therefore, we chose to also test a hybrid of the two formulations. The hybrid formulation is a limited version of the time-indexed formulation, where only a reduced set of the violated cuts are added. More specifically, we only add cuts if they are maximally violated, i.e. if the left hand side of constraint (28) is equal to 2.

First, to give an idea of the overall performance of the three approaches, the test results are summarized in Table 3. Secondly, to give a more detailed view, we present some specific analyses of various problem properties. As the number of temporal dependencies increases, the problem becomes more constrained, and hence the value of the optimal solution will increase with the number of dependencies. It is interesting to look at the lower bound found in the root node of the branch-and-bound tree. In a straight forward version of the relaxed formulation, no generalized precedence constraints have

	Instances in total	Time-indexed formulation (all cuts) Solved in root	Time-indexed formulation (all cuts) Solved before timeout	Time-indexed formulation (limited) Solved in root	Time-indexed formulation (limited) Solved before timeout	Relaxed formulation Solved in root	Relaxed formulation Solved before timeout
Synchronization	1148	483	1027	448	1141	138	1143
Overlap	1324	351	1058	322	1207	127	1240
Minimum difference	1400	741	1350	703	1377	226	1381
Min+max difference	1400	531	1226	465	1361	105	1384
Random mix	1400	506	1271	459	1382	155	1383

Table 3: Overview of the test results.

any effect in the root node, and the lower bound, naturally, is unchanged. Using the time window reductions, we are, however, able to remove parts of the solution space and this may increase the lower bound. The time-indexed formulation is always stronger and hence the lower bounds of this formulation are always greater than or equal to the ones of the relaxed formulation. For the same reason, the lower bound of the limited version will always lie between the lower bounds of the other two. This is illustrated for instance `R110` in Figure 6.

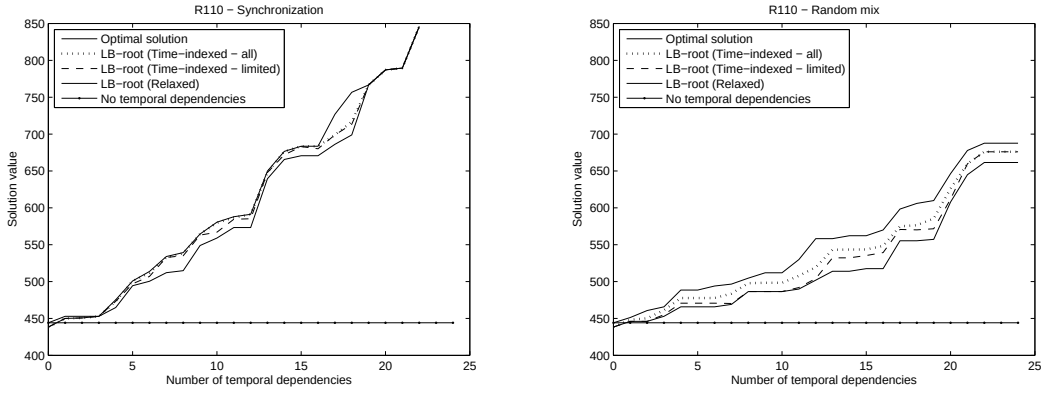

Figure 6: Comparison of solution values and lower bounds of the branch-and-bound root node.

Comparing the two graphs of Figure 6, we also see that the synchronization constraints are stricter than random dependencies and therefore the optimal solution value increases faster. The tendency observed, for instance `R110`, is consistent with the general picture.

As shown in Figure 6, the root node lower bound sometimes coincides with the value

of the optimal solution. In such cases, we often find the optimal solution in the root node. Since this will result in low computation times, it is interesting to see how often it happens.

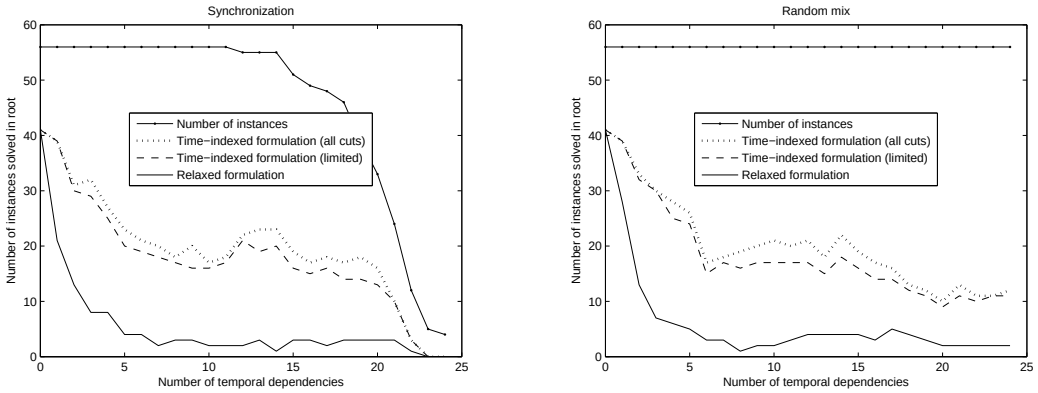

Figure 7: Number of instances solved in the root node of the branch-and-bound tree.

In Figure 7 the total number of instances solved in the root node is given, summarized over all 56 instances. Again, we clearly observe the strength of the time-indexed formulation. There is a significant increase in the number of instances solved in the root node compared to the relaxed formulation. Interestingly, there is not much difference from the full formulation to the limited version. In the relaxed formulation, if a problem can be solved in the root node, it means that all temporal dependencies were respected by chance, and hence they would not have been very constraining. Figure 8 gives the number of nodes in the branch-and-bound tree (the mean over all instances) and the conclusions are the same as for Figure 7. As we would expect for the relaxed formulation, we see that the number of nodes increases with the number of temporal dependencies. This does not seem to be the case for the time-indexed formulation.

Another interesting aspect is the solution time. We examine the solution time for each of the instances individually and also consider the general trend. The variation on solution time is large between the instances. This means that taking an average of these values would emphasize the harder instances, but we want them to count equally in this test. Therefore, we normalize the values by comparing each computation time to the solution time for the same problem without temporal dependencies. The mean over all instances is shown in Figure 9.

Looking at Figure 9, it is clear that the time-indexed formulation is worse than the other two with respect to solution time. Closer inspection shows that the full time-indexed formulation has a few instances where computation time is excessive and this has a major impact on the mean value.

In connection with solution time, it is also very interesting to make a direct comparison between the approaches for each instance. For each number of temporal dependencies, we count the number of instances where the limited time-indexed approach is faster

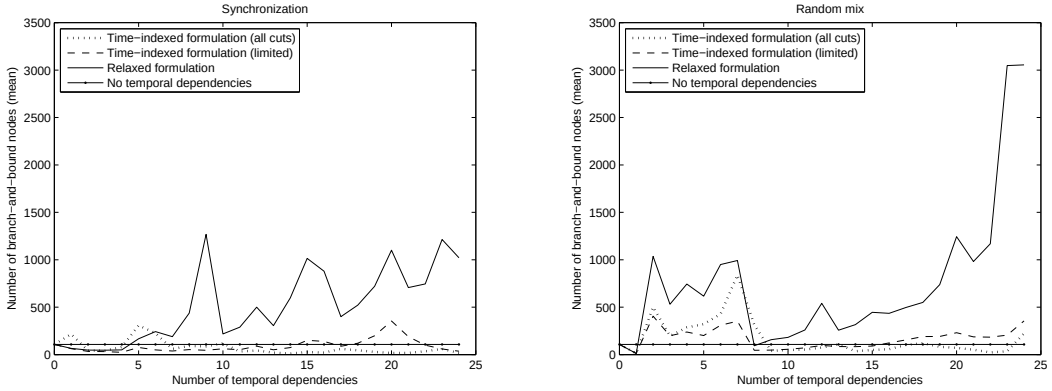

Figure 8: Number of nodes in the branch-and-bound tree.

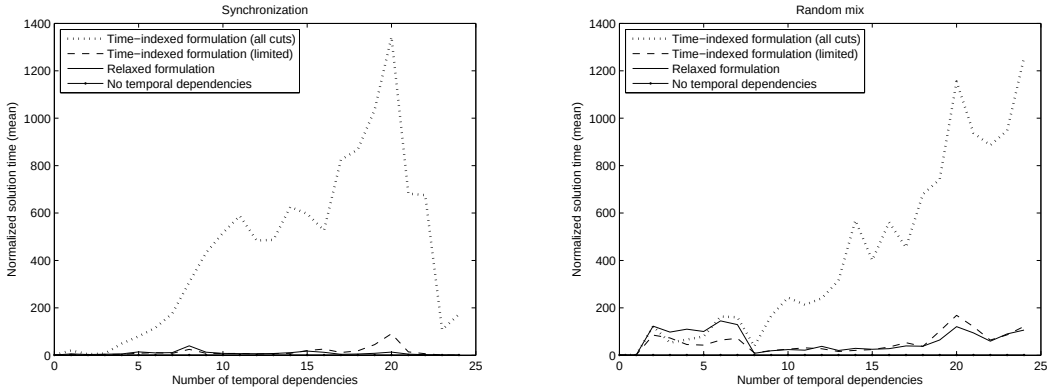

Figure 9: Normalized solution time (mean).

than the relaxed formulation and vice versa. The results are summarized in Figure 10. Looking at the instances individually, the limited time-indexed approach seems best.

Finally, we look at the distribution of time spent in the algorithm. This is illustrated in Figure 11. For the time-indexed formulation, the portion of time spent in the LP-solver increases as problems with more temporal dependencies are considered. This is due to the fact that more cuts are added for these problems and hence the size of the LP-model increases. For the relaxed formulation, the tendency is not surprisingly that more time is spent branching when the number of temporal dependencies increases. The share of time spent by the LP-solver is, in this case, stable.

On the basis of the tests, we are able to conclude that the temporal dependencies introduce additional complexity to the problem, as expected. The time-indexed formulation has the worst immediate performance, but may be more useful in large problems with harder pricing problems. The limited time-indexed approach seems to have the

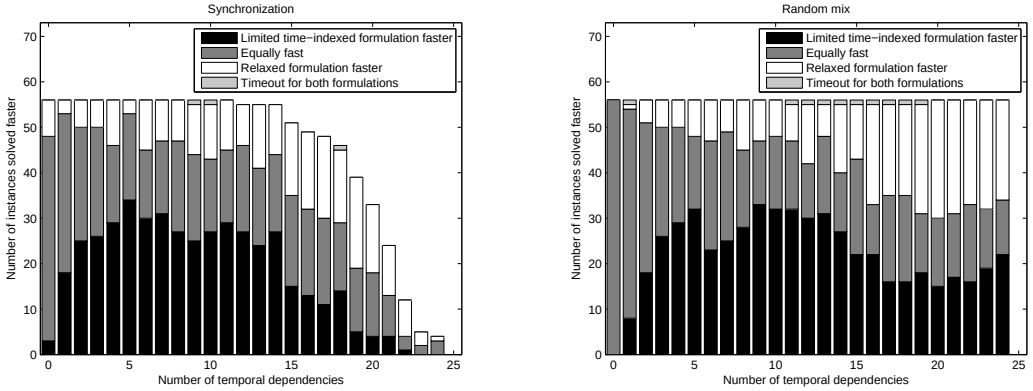

Figure 10: Number of tests where one of the approaches is faster than the other. The two approaches are considered equally fast if they are within 20% of each other.

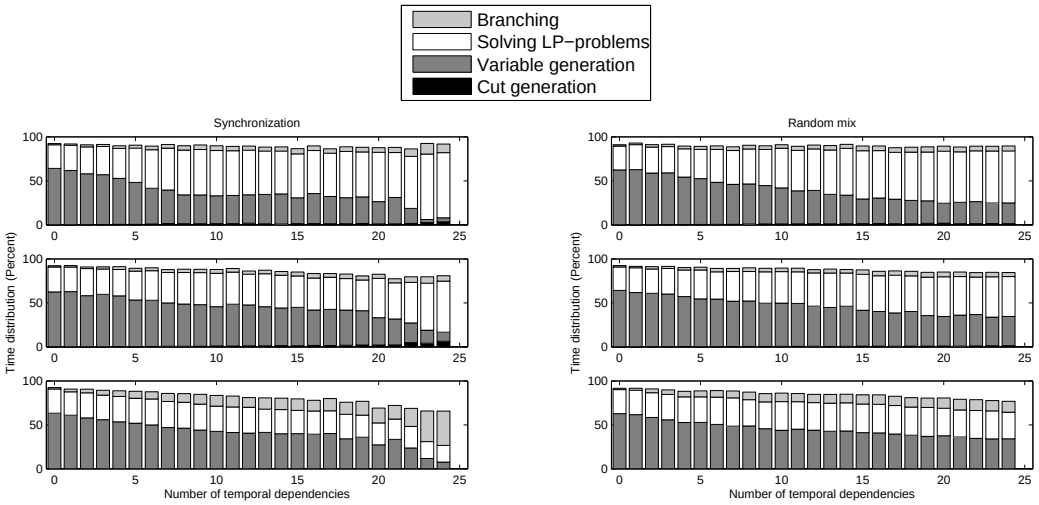

Figure 11: Distribution of solution time for the time-indexed model (top), the limited time-indexed model (middle), and the relaxed model (bottom).

best performance in our tests, but the relaxed formulation is not far behind. A few instances of each type turn out to be very hard to solve, no matter what method is used. The time-indexed formulation does have a number of nice features that could be utilized in future development. It has tighter bounds, both theoretically and in the practical instances that we have examined. The tighter bounds mean that more instances are solved in the root node of the branch-and-bound tree, and in these cases the formulation gives better results. Also, for instances where the solution is not found in the root node, the branch-and-bound tree is still significantly smaller than the corresponding tree

for the relaxed formulation. The number of variables that has to be generated is also generally smaller for the time-indexed formulation. For most realistic problems, variable generation is the dominating factor of the overall solution time, and in these cases the time-indexed formulation may be the better choice.

6 Conclusions and future work

The vehicle routing problem with time windows and temporal dependencies has been introduced. The problem has previously been treated in various practical contexts in different forms, but this is the first generic analysis presented in the literature. Four different models were presented and ranked according to their theoretical strength. The time-indexed model has the tightest formulation and hence gives the best bounds, but the number of constraints is so large that they cannot be included explicitly. Instead, the model was implemented in a branch-and-cut-and-price context, where both constraints and variables are generated dynamically. As this approach is novel, a separation algorithm was presented and the necessary adjustments in the pricing problem were introduced. The branching scheme was presented next. The scheme is based on the traditional time window branching, but it is also used to restore feasibility with respect to temporal dependencies. The branching scheme is as strong as the previously presented branching scheme for problems with synchronization constraints only. Finally, the benchmark instances were introduced and a quantitative analysis was carried out.

The analysis showed that, even though the time-indexed model has some nice properties, it also retains its major drawback, namely the number of constraints. As a consequence, a hybrid method was implemented, where only a limited number of the violated cuts are added. This approach kept most of the nice features of the time-indexed model, while at the same time lowering the solution time to the same level as the solution time of the relaxed model. In fact the hybrid method is only slower than the relaxed model in a small number of instances.

The model presented in this paper is general and is therefore applicable to various practical problems. Future work could be adaption to real world problems. Another very interesting direction for future research could be to include additional cuts. Using the time-indexed formulation, we were able to solve many instances already in the root node of the branch-and-bound tree, and this number could be increased by introducing additional cuts. The performance of the time-indexed model was clearly better than the relaxed model for the instances where the optimal solution was obtained in the root node.

References

T. Achterberg, T. Koch, and A. Martin. Branching rules revisited. *Operations Research Letters*, 33(1):42–54, 2005.

N. Bélanger, G. Desaulniers, F. Soumis, and J. Desrosiers. Periodic airline fleet assignment with time windows, spacing constraints, and time dependent revenues. *European Journal of Operational Research*, 175(3):1754–1766, 2006. ISSN 03772217.

D. Bredström and M. Rönnqvist. A branch and price algorithm for the combined vehicle routing and scheduling problem with synchronization constraints. Discussion Paper 2007/7, Norwegian School of Economics and Business Administration - Department of Finance and Management Science, Norway, 2007.

D. Bredström and M. Rönnqvist. Combined vehicle routing and scheduling with temporal precedence and synchronization constraints. *European Journal of Operational Research*, 191(1):19–31, 2008. ISSN 03772217.

A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computers and Operations Research*, 33(10):2972–2990, 2006.

M. Christiansen and B. Nygreen. *Robust Inventory Ship Routing by Column Generation*, chapter 7, pages 197–224. Desaulniers G., Desrosiers J., Solomon M.M.: Column Generation, Springer, New York, 2005.

Coin. COmputational INfrastructure for Operations Research (COIN-OR), 2006. http://www.coin-or.org/.

W. Cook and J. L. Rich. A parallel cutting-plane algorithm for the vehicle routing problem with time windows. Technical report, Rice University, 2001.

E. Danna and C. L. Pape. *Branch-and-Price Heuristics: A Case Study on the Vehicle Routing Problem with Time Windows*, chapter 4, pages 99–129. Desaulniers G., Desrosiers J., Solomon M.M.: Column Generation, Springer, New York, 2005.

G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8(1):101–111, 1960.

G. Desaulniers, F. Lessard, and A. Hadjar. Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. *Transportation Science*, 42(3):387, 2008. ISSN 00411655.

M. Desrochers, J. Desrosiers, and M. M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40:342–354, 1992.

K. F. Doerner, M. Gronalt, R. F. Hartl, G. Kiechle, and M. Reimann. Exact and heuristic algorithms for the vehicle routing problem with multiple interdependent time windows. *Computers and Operations Research*, 35(9):3034–3048, 2008. ISSN 03050548.

A. Dohn, M. S. Rasmussen, T. Justesen, and J. Larsen. The home care crew scheduling problem. In K. Sheibani, editor, *ICAOR'08 - Proceedings, 1st International Conference on Applied Operational Research*, pages 1–8. Tadbir Institute for Operational Research, 2008.

A. Dohn, E. Kolind, and J. Clausen. The manpower allocation problem with time windows and job-teaming constraints: A branch-and-price approach. *Computers and Operations Research*, 36(4):1145–1157, 2009a.

A. Dohn, M. S. Rasmussen, and J. Larsen. Technical report: The vehicle routing problem with time windows and temporal dependencies. Technical report, Department of Management Engineering, Technical University of Denmark, Kgs. Lyngby, Denmark, 2009b.

M. Dror. Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research*, 42(5):977–978, 1994.

D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.

A. Fügenschuh. The vehicle routing problem with coupled time windows. *Central European Journal of Operations Research*, 14(2):157–176, 2006. ISSN 1435246x.

S. Gélinas, M. Desrochers, J. Desrosiers, and M. M. Solomon. A new branching strategy for time constrained routing problems with application to backhauling. *Annals of Operations Research*, 61:91–109, 1995.

I. Ioachim, S. Gelinas, F. Soumis, and J. Desrosiers. A dynamic programming algorithm for the shortest path problem with time windows and linear node costs. *Networks*, 31 (3):193–204, 1998. ISSN 00283045.

I. Ioachim, J. Desrosiers, F. Soumis, and N. Belanger. Fleet assignment and routing with schedule synchronization constraints. *European Journal of Operational Research*, 119 (1):75–90, 1999.

M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56(2): 497–511, 2008.

T. Justesen and M. S. Rasmussen. The home care crew scheduling problem. Master's thesis, Informatics and Mathematical Modeling, Technical University of Denmark, 2008.

B. Kallehauge, J. Larsen, O.B.G. Madsen, and M.M. Solomon. *Vehicle Routing Problem with Time Windows*, chapter 3, pages 67–98. Desaulniers G., Desrosiers J., Solomon M.M.: Column Generation, Springer, NY, 2005.

P. Kilby, P. Prosser, and P. Shaw. A comparison of traditional and constraint-based heuristic methods on vehicle routing problems with side constraints. *Constraints*, 5 (4):389–414, 2000.

N. Kohl, J. Desrosiers, O. B. G. Madsen, M. M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1): 101–116, 1999.

D. Lesaint, N. Azarmi, R. Laithwaite, and P. Walker. Engineering dynamic scheduler for work manager. *BT Technology Journal*, 16(3):16–29, 1998.

Y. Li, A. Lim, and B. Rodrigues. Manpower allocation with time windows and job-teaming constraints. *Naval Research Logistics*, 52:302–311, 2005.

A. Lim, B. Rodrigues, and L. Song. Manpower allocation with time windows. *Journal of the Operational Research Society*, 55:1178–1186, 2004.

R. Lougee-Heimer. The common optimization INterface for operations research: Promoting open-source software in the operations research community. *IBM Journal of Research and Development*, 47(1):57–66, 2003.

J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100(2):423–445, 2004.

D. Oron, S.-N. Sze, and A. S.-F. Ng. A heuristic manpower scheduling for in-flight catering service. The 13th International Conference of Hong Kong Society for Transportation Studies, 2008.

L.-M. Rousseau, M. Gendreau, and G. Pesant. The synchronized vehicle dispatching problem. Technical Report CRT-2003-11, Centre de Recherche sur les Transports, Université de Montréal, Canada, 2003. Conference paper, Odysseus 2003.

D. M. Ryan. The solution of massive generalized set partitioning problems in aircrew rostering. *Journal of the Operational Research Society*, 43(5):459–467, 1992. ISSN 01605682.

M.W.P. Savelsbergh. Local search in routing problems with time windows. *Annals of Operations Research*, 4(1-4):285–305, 1985. ISSN 02545330.

M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987.

J.M. Van Den Akker, J.A. Hoogeveen, and J.W. Van Kempen. Parallel machine scheduling through column generation: Minimax objective functions. *Lecture Notes in Computer Science*, 4168:648–659, 2006. 14th Annual European Symposium on Algorithms, ESA 2006.

In this paper, we formulate the vehicle routing problem with time windows and temporal dependencies. The problem is an extension of the well studied vehicle routing problem with time windows. In addition to the usual constraints, a scheduled time of one visit may restrain the scheduling options of other visits. Special cases of temporal dependencies are synchronization and precedence constraints. Two compact formulations of the problem are introduced and the Dantzig-Wolfe decompositions of these formulations are presented to allow for a column-generation-based solution approach. Temporal dependencies are modeled by generalized precedence constraints. A total of four different master problem formulations are proposed and it is shown that the formulations can be ranked according to the tightness with which they describe the solution space.
A tailored time window branching is used to enforce feasibility on the relaxed master problems. Finally, a computational study is carried out to quantitatively reveal strengths and weaknesses of the proposed formulations. It is concluded that, depending on the problem at hand, the best performance is achieved either by relaxing the generalized precedence constraints in the master problem, or by using a time-indexed model, where generalized precedence constraints are added as cuts when they become severely violated.

ISBN 978-87-90855-36-9

DTU Management Engineering
Department of Management Engineering
Technical University of Denmark

Produktionstorvet
Building 424
2800 Kongens Lyngby
Tel. 45 25 48 00
Fax 45 93 34 35

www.man.dtu.dk